The Security Aspects of On-Line Science[1]

William E. Johnston[2]
Information and Computing Sciences Division
Ernest Orlando Lawrence Berkeley National Laboratory
University of California

[1] This work is supported by the Director, Office of Energy Research, Office of Computation and Technology Research, Mathematical, Information, and Computational Sciences Division, of the U. S. Department of Energy under Contract No. DE-AC03-76SF00098 with the University of California

[2] wejohnston@lbl.gov (510-486-5014) - http://www-itg.lbl.gov

Motivation

- ♦ When we provide a “security” capability it should come with a vulnerability analysis as well as a feature list
- ♦ Threat analysis is also important, but much harder to come by (classically the purview of the intelligence community)
    - a vulnerability without a threat is not a risk (today!)
- ♦ A “disinterested” third-party should do the vulnerability analysis

DOE2000 Diesel Collaboratory as Prototype for A “Secure” On-Line Scientific Community

- Small-medium community (~ 50-100 people)
- Dozen organizations (DOE Labs, Universities, Industry)
- Trust relationships - who gets to see what and when - already established
- Typical concerns:
    - proprietary data (industry)
    - confidential data (unpublished)
    - open data
    - many fluid and overlapping access groups (different groups for different data at different times)
- On-line sharing currently minimal (no security)

- Individual datasets probably hard to valuate
  (a new diesel engine design may be initially speculative, but very valuable after modeling and testing proves the concept)
- Many different capabilities need to be protected
    - data in various stages of analysis
    - modeling codes and their execution environments
    - video/audio/whiteboard conferencing
    - shared workspaces
    - e-mail
    - John Howard, SNL, is producing a comprehensive security requirements document

On-Line System Security

(adapted from Steve Kent’s NDSS talk “Network Security Principles”)

Security Terminology

- Vulnerabilities
  - security flaws in systems
- Attacks
  - means of exploiting vulnerabilities
  - a means of attack is benign without a threat
- Threats
  - motivated adversaries capable of mounting attacks which exploit vulnerabilities

Terminology

- Risks
  - consequences of a successful attack *including* the “value” of the compromised resource
  - disclosure of information
  - corruption of information
  - denial of service
  - theft of service
- Countermeasures
  - technical or procedural means of addressing vulnerabilities or thwarting specific attacks

The Threat: Motivated Adversaries (The Bad Guys)

- Hackers
  - ever present, motivated by bragging rights or “revenge”
  - potential partners of more serious bad guys (a concern in the cyberwarfare community)
- Disgruntled employees
  - insider compromise accounts for something like 70% of all successful attacks in the financial arena
- Industrial or Academic spies
  - motivated by financial or status gain

Threat

- Terrorists
  - will have broad, well funded, well equipped capabilities that attack the whole range of supporting services (personnel, systems, infrastructure)
    - the last significant penetration of our systems involved the successful telephone impersonation of an LBNL employee (but the attacker was a game player, not a terrorist)
  - we tend to dismiss this threat because most current such adversaries are technologically unsophisticated - this is probably a big mistake, because when such an attack does come, it will be devastating
- Special interest groups
- Journalists

Threat

- ♦ **“Real” spies**
- ♦ **Criminals (organized or otherwise)**

Security Continuum

- ♦ **There are no perfect, secure systems**
- ♦ **Systems are “adequately secure” only relative to a perceived threat**
  - • **amateur hackers (opportunistic)**
  - • **skilled invaders that target you, and whom you will not detect, and who will set up disguised information feeds or time bombs that will be effective over a long period of time**
  - • **targeted attacks by skilled perpetrators are difficult, if not impossible, to defend against**

Reality

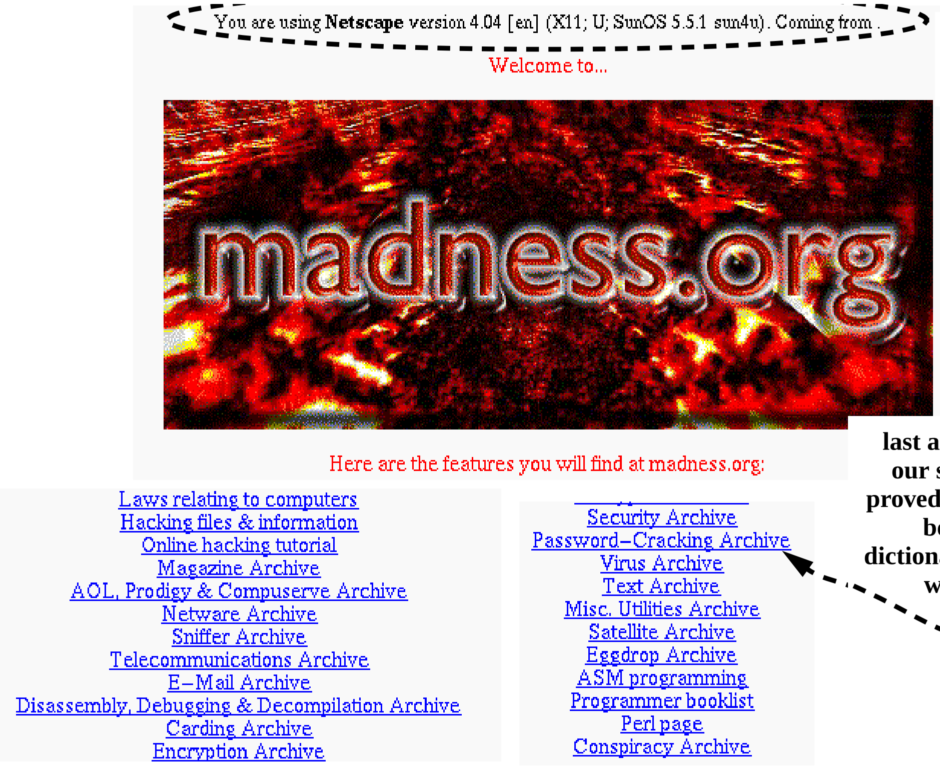
- ♦ **Risk analysis, if properly performed, provides a methodology for identifying what constitutes *adequate* security**
  - **identify and valuate resources, analyze consequences, then design protection strategies accordingly (“rarely done completely or correctly” - SK)**

The Threshold Effect (“Script Kiddies”)- Probably Biggest Current Problem

- ♦ **Experts are putting powerful attack tools into the hands of amateurs: Once a technical attack against a security technology has been “debugged” it can be executed by a wide range of (inexperienced) attackers**
- ♦ **Thus it is dangerous to dismiss a attack as “too complex or too technical” because you think that the perceived attackers do not possess the technical capability to mount the attack.**
- ♦ **www.rootshell.com, www.madness.org (a cyber-defence concern is the growing worldwide pool of hackers)**

  From: "Andrejs Bojars" of Latvia: aga, skaisti, buus laikam jaasagraabj :) no savas puses varu iedoe teel dazhus jaukus linkus no mana bookmark faila: http://l0pht.com, http://www.paranoia.com/~coldfire/index.html, http://www.cdwarez.com, http://ritalin.shout.net/~anemic/hotlist.html, http://www.intersurf.com/~materva/files.html, http://www.2600.com/, ...

Current Attacks



You are using **Netscape** version 4.04 [en] (X11; U; SunOS 5.5.1 sun4u). Coming from .

Welcome to...

Here are the features you will find at madness.org:

Laws relating to computers
Hacking files & information
Online hacking tutorial
Magazine Archive
AOL, Prodigy & Compuserve Archive
Netware Archive
Sniffer Archive
Telecommunications Archive
E-Mail Archive
Disassembly, Debugging & Decompilation Archive
Carding Archive
Encryption Archive

Security Archive
Password-Cracking Archive
Virus Archive
Text Archive
Misc. Utilities Archive
Satellite Archive
Eggdrop Archive
ASM programming
Programmer booklist
Perl page
Conspiracy Archive

just to get your attention

last attack on our systems proved that they better dictionaries than we do

Current Attacks

aix_ping.c -Overwrites a buffer in gethostbyname(), giving root on AIX 4.x PPC systems. -
aix_lchangelv.c -Another buffer overrun exploit that gives root on AIX 4.x PPC from lchangelv. -
aix_xlock.c -This will overwrite a buffer in /usr/bin/X11/xlock on AIX 4.x PPC, giving root. -
web_sniff.c -A Linux sniffer that is designed to retrieve web usernames and passwords. -
xf86_ports.txt -A normal user can run X on a reserved port thus blocking legitmate daemons. -
solaris_telnet.c -A program designed to attack a Solaris 2.5 box, making it totally unresponsive. -
identd_attack.txt -A massive amount of authorization requests can render a system unusable. -
secure_shell.txt -Using SSH, a non-root user can open privleged ports and redirect them. -
sshd_redirect.txt -Any normal user can redirect privileged ports using secure shell daemon. -
bsd_procfs.c -In /proc under FreeBSD 2.2.1, you can modify a setuid executable’s memory. -
zgv_exploit.c -This will overwrite a buffer in /usr/bin/zgv on Redhat Linux systems, giving root. -
sgi_html.txt -It is possible to execute remote commands on IRIX 6.3 and 6.4 via /usr/sysadm. -
smurf.c -Spoofs IMCP packets resulting in multiple replies to a host from a single packet. -
in.comstat.txt -If a user has biff y on, in.comstat can be used increase the system load. -
bind_nuke.txt -Bind8.1.(1) can’t update the same RR more than once in the same DNS packet. -
chkexploit_1.13.tgz -A shell script for Linux that checks for some publicly available exploits. -
syslog_deluxe.c -Lets you write spoofed and arbitrary messages to another machine’s syslogd. -
dgux_fingerd.txt -The fingerd that ships w/ dgux allows remote execution of arbitrary commands. -
smb_mount.c -This overwrites a buffer on Linux systems in smbmount from smbfs-2.0.1. -
nmap.1.25.tar.gz -nmap is a utility for port scanning large networks and currently runs on Linux. -
innd_exploit.c -Overwrites a buffer in innd on Linux x86 systems thus giving a remote shell. -
smlogic.c -This is a fully functional logic bomb designed render Linux systems unuseable. -
ld.so.c -Overwrites a buffer via LD_PRELOAD env. variable, giving root on Linux. -
promisc.c -This program will scan your network devices to detect running sniffers. -
solaris_ping.txt -On Solaris 2.x systems, any user can crash or reboot the system using ping. -
seyon_exploit.sh -Exploit for seyon, giving you the euid or egid of whatever seyon is suid to. -
aixdtaction.c -Overwrites a buffer in /usr/dt/bin/dtaction via HOME env. variable, giving root. -
datapipe.c -Makes a pipe between a listen port on localhost and a port on a remote machine. -
sping.tar.gz -Linux binary and source of ‘sping’ which causes Win95 machines to crash. -
linux_httpd.c -Overwrites a buffer in NSCA httpd v1.3 on linux systems, giving a remote shell. -

Current Attacks

♦ Recent Teardrop2 attack disabled (to the extent of requiring manual intervention to restart the system) probably 10,000 computers at several dozen US Gov. and University sites

- several suspected motivations, including bragging rights (vis a vis Gates Congressional testimony) or revenge (for the arrest of the Pentagon hackers the previous week)
- correcting this problem can take an hour or more per system (to install the necessary patches)

♦ Excellent summary of hacker techniques (tutorial notes from last Usenix Security Conference):

“Network Security Profiles: *What Every Hacker Already Knows About You, and How They Do It*”
Brad.Johnson@SystemExperts.com and
Jon.Rochlis@SystemExperts.com
http://www.systemexperts.com/Classes.htm

Sources of Vulnerabilities

Vulnerabilities are just flaws that are security relevant

♦ Design flaws
- infrastructure, operating system & application vulnerabilities
  - Teardrop: The OS overwrites itself
  - panix.com - syn-flood denial of service
- protocol design vulnerabilities (an application operates correctly to do something bad)

♦ Implementation flaws
- programming errors
- undocumented system & application “features”

Vulnerabilities

♦ **Mismanagement**

- **unintended and/or residual authorizations**
- **failure to deploy security bug fixes (however, this could be the result of a risk analysis)**
    - **We had a student who took down a freeBSD system and brought it up as a RedHat Linux system from the CD. Within two hours external hackers had used an IMAP server bug to gain root access.**
    - **“If you aren’t attacked (or at least probed) within an hour of connecting to the net, you should probably contact your ISP - because you probably aren’t really connected!” - SK**

Vulnerabilities

Most systems are mis-configured / mis-managed from a security point of view (it costs too much money to do it right, and there is no perceived benefit to the end user - therefore little motivation).

One appeal of firewalls is that there is a single point of management for security - most organizations have given up trying to manage the thousands of systems behind the firewall. However, firewalls bring their own set of problems.

Up Front Issues

- ♦ Security policy should match risk and practice whenever possible

  The most valuable data should be protected with "best current practice in open environments"

  (However, few scientific communities have the resources to protect anything as classified data would probably have to be protected in open networks. Nor are they probably serious targets.)

- ♦ "We will just keep all of our valuable data off of the net."

  This is an unrealistic assumption. The economics of global enterprise of any description dictate the use of commodity telecommunications infrastructure. (E.g., in Desert Storm, most Army logistics was done over the Internet, and most voice and fax communication used the commercial infrastructure.)

Diesel Collaboratory Approach

Hypothesis: Remotely accessed bastion hosts with strong user authentication can provide "good" security. (In fact, probably better than locally accessed data on private systems.)

- ♦ Medium grained ("directory-level") access control based on remotely managed, "strong" authorization and user credentials and trusted third-parties
  - this represents "pretty strong" *application-level* security
- ♦ Use "bastion" hosts for servers to reduce the number of vulnerabilities
  - Certification Authority (user credential issuer)
  - CA directory server
  - confidential and public data on separate systems

Diesel Collab.

- ♦ Vulnerabilities of PKI access control (strengths in the Akenti access control system talk)
  - mis-identification of principals
    - CA identity policy should follow community practice (too elaborate a policy for the resource being protected will inhibit deployment)
      - + Put minimum policy in the CA certs!! Use authorization credentials for specific policy - WEJ
    - CAs should just certify namespaces for which they are authoritative, then there is no issue as to their authority to provide the name-key binding
    - Cross certificating a dozen organization’s CA’s would be impossible

      (however Akenti does not need such cross-certification

Diesel Collab.

because the namespace of all users (i.e. their CA) is given explicitly, and authorization policy is separate from identity policy)

- Diesel CA - a “private community policy” CA
  - fax request of cert. on org. letterhead
  - policy-based review of appropriateness
  - “reverse lookup” by obtaining telephone number from organizational source (e.g. personnel dept.), followed by telephone contact
  - issue cert. or accept existing cert (i.e., impose Diesel community policy on top of org. CA policy)

- mis-specification of access requirements
  - human error in formulating use-conditions / authorizations
- untrustworthy “trusted” third-parties
  - insider attacks (“root” on most any system can compromise data or resources)

Diesel Collab.

- denial of service through unavailability of certs.
  - largely an infrastructure issue that must be addressed by, e.g., IPv6 security and secure DNS.
- server penetration
  - data is not, but could be, encrypted and authenticated (with additional level of complexity)
- Self signed CA certificate
  - this makes the X.509 (identify certificate) directory vulnerable to bogus certificate attack
  - We need a “light-weight” CA hierarchy whose only purpose is to certify the name (and maybe organizational home) of a CA
- CA root key compromise
  - bogus certificates, even with the CA cert. signed

Diesel Collab.

- Subject key compromise
  - most likely - user exposes private-key passphrase
  - can be hard to detect (have to analyze audit logs for out-of-norm access behavior)
  - an advantage of physical tokens (e.g. crypto SmartCards) is that you know when you have lost it